Formatos de Instrucción

Para empezar a diseñar un microprocesador debemos especificar el formato de las instrucciones. Este formato representa la forma en que cada instrucción es almacenada en la memoria de programa del microprocesador.

El conjunto de instrucciones del miniESCOMIPS tienen un **formato de 15 bits**, con el que podemos realizar instrucciones con 3 operandos. Básicamente se tienen 3 formatos de instrucciones que son: **Formato tipo R, formato tipo I y formato tipo J**. Estos formatos de instrucción permiten manejar los modos de direccionamiento inmediato, por registro, directo e indirecto.

Formato de Instrucción tipo R (Register):

Este formato lo tienen todas las instrucciones cuyos operandos son todos registros. Los 15 bits de la instrucción se distribuyen en 5 campos de la siguiente manera:

14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OPCODE					Rd		Rs1		Rs2		FUNCTION			
5 bits					2 bits		2 bits		2 bits		4 bits			

El significado de cada uno de los campos es:

OPCODE: Es el código de operación. En las instrucciones Tipo R el código de operación siempre es CERO.
Rd: Es el registro operando destino.
Rs1: Es el primer registro operando fuente.
Rs2: Es el segundo registro operando fuente.
FUNCTION: Este campo se le denomina código de función. Este código permite distinguir el tipo de instrucción a ejecutar.

Formato de Instrucción tipo I (Inmediate):

Este formato lo tienen todas las instrucciones donde uno de los operandos es un número de 8 ó 6 bits que representa una constante o dirección. Los 15 bits de la instrucción se distribuyen en 3 campos cuando la constante o dirección es de 8 bits y en 4 campos cuando es de 6 bits, tal como se muestra a continuación:

14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OPCODE					Rd		Constante o dirección							
5 bits					2 bits		8 bits							

El significado de cada uno de los campos es:

OPCODE: Es el código de operación.
Rd: Es el registro operando destino.

AUTOR: VICTOR HUGO GARCÍA ORTEGA

Constante: Dato de 8 bits que representa un número inmediato o dirección.

14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OPCODE					Rd		Rs1		Constante o dirección					
5 bits					2 bits		2 bits		6 bits					

El significado de cada uno de los campos es:

OPCODE: Es el código de operación.
Rd: Es el registro operando destino.
Rs1: Es el primer registro operando fuente
Dirección: Dato de 6 bits que representan un número inmediato o dirección.

Formato de Instrucción tipo J (Jump):

Este formato lo tienen las instrucciones de salto. Los 15 bits de la instrucción se distribuyen en 3 campos, tal como se muestra a continuación:

14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OPCODE					Sin uso		Constante o dirección							
5 bits					2 bits		8 bits							

El significado de cada uno de los campos es:

OP: Es el código de operación.
Constante o dirección: Representa la dirección de 8 bits en las instrucciones de salto.

Con estos formatos de instrucción se diseñan todas las instrucciones del microprocesador. Dichas instrucciones se describen a continuación.

CONJUNTO DE INSTRUCCIONES.

Las instrucciones propuestas en el ESCOMIPS son instrucciones que se pueden agrupar en las siguientes categorías:

- Instrucciones de carga y almacenamiento
- Instrucciones aritméticas
- Instrucciones lógicas
- Instrucción de salto incondicional.
- Instrucciones de comparación.
- Instrucciones de saltos condicionales.
- Instrucciones de otro tipo.

Estas instrucciones se muestran en la tabla 1.

INSTRUCCIONES DE CARGA Y ALMACENAMIENTO									
Instr.	Ejemplo	Significado	Formato de instrucción					Formato	Banderas
LI	LI Rd, #Slit8	Rd = Slit8	01	Rd	Slit8			I	Ninguna
LWI	LWI Rd, lit8	Rd = Mem[lit8]	02	Rd	lit8			I	Ninguna
SWI	SWI Rd, lit8	Mem[lit8] = Rd	03	Rd	lit8			I	Ninguna
INSTRUCCIONES ARITMÉTICAS									
ADD	ADD Rd,Rs1,Rs2	Rd = Rs1+Rs2	00	Rd	Rs1	Rs2	00	R	C N Z OV
SUB	SUB Rd,Rs1,Rs2	Rd = Rs1-Rs2	00	Rd	Rs1	Rs2	01	R	C N Z OV
ADDI	ADDI Rd,Rs1,#Slit6	Rd = Rs1+Slit6	04	Rd	Rs1	Slit6		I	C N Z OV
SUBI	SUBI Rd,Rs1,#Slit6	Rd = Rs1-Slit6	05	Rd	Rs1	Slit6		I	C N Z OV
INSTRUCCIONES LÓGICAS									
AND	AND Rd,Rs1,Rs2	Rd = Rs1 & Rs2	00	Rd	Rs1	Rs2	02	R	N Z
OR	OR Rd,Rs1,Rs2	Rd = Rs1 | Rs2	00	Rd	Rs1	Rs2	03	R	N Z
XOR	XOR Rd,Rs1,Rs2	Rd = Rs1 ^ Rs2	00	Rd	Rs1	Rs2	04	R	N Z
NAND	NAND Rd,Rs1,Rs2	Rd = ~(Rs1 & Rs2)	00	Rd	Rs1	Rs2	05	R	N Z
NOR	NOR Rd,Rs1,Rs2	Rd = ~(Rs1 | Rs2)	00	Rd	Rs1	Rs2	06	R	N Z
XNOR	NOR Rd,Rs1,Rs2	Rd = ~(Rs1 ^ Rs2)	00	Rd	Rs1	Rs2	07	R	N Z
INSTRUCCIÓN DE SALTO INCONDICIONAL									
B	B lit8	PC = lit8	06	S/U	lit8			J	NINGUNA
INSTRUCCIONES DE COMPARACIÓN									
CP	CP Rs1, Rs2	Rs1 – Rs2 Reg_edo = Flags	00	S/U	Rs1	Rs2	08	R	C N Z OV
CPI	CPI Rd, #Slit8	Rd – Slit8	07	Rd	Slit8			I	C N Z OV

AUTOR: VICTOR HUGO GARCÍA ORTEGA

		Reg_edo = Flags						
INSTRUCCIONES DE SALTOS CONDICIONALES								
BEQ	BEQ lit8	If(EQ) PC = lit8	08	S/U	lit8		J	Ninguna
BNEQ	BNEQ lit8	If(NEQ) PC = lit8	09	S/U	lit8		J	Ninguna
BLT	BLT lit8	If(LT) PC = lit8	10	S/U	lit8		J	Ninguna
BLET	BLET lit8	If(LET) PC = lit8	11	S/U	lit8		J	Ninguna
BGT	BGT lit8	If(GT) PC = lit8	12	S/U	lit8		J	Ninguna
BGET	BGET lit8	If(GET) PC = lit8	13	S/U	lit8		J	Ninguna
OTRAS INSTRUCCIONES								
NOP	NOP	No operation	14	S/U	S/U	S/U	S/U		Ninguna

Tabla 1 Instrucciones del miniESCOMIPS

AUTOR: VICTOR HUGO GARCÍA ORTEGA

Instrucciones de carga y almacenamiento.

Estas instrucciones se encargan de transferir datos de la memoria a los registros y viceversa. La instrucción LI carga un valor inmediato a cualquier registro. Las instrucciones LWI y SWI cargan un valor de memoria a registro y registro a memoria respectivamente. Estas instrucciones soportan direccionamiento inmediato.

Instrucciones aritméticas

Estas instrucciones consisten en las operaciones aritméticas suma y resta. Estas instrucciones soportan direccionamiento inmediato y por registro.

Instrucciones lógicas

Estas instrucciones consisten en las operaciones lógicas: AND, OR, XOR, NAND, NOR y XNOR. Estas instrucciones soportan direccionamiento por registro.

Instrucción de brinco incondicional.

La instrucción de brinco incondicional (*B)* realiza el salto a la dirección de memoria de 8 bits especificada en la instrucción.

Instrucciones de comparación.

Estas instrucciones permiten comparar dos números los cuales son los operandos de la instrucción. La comparación consiste en realizar una operación de resta entre los dos operandos, al realizar la resta se obtiene un resultado en las banderas N, Z, C y OV de la ALU. Estas banderas con almacenadas en el Registro de Banderas o Registro de Estado del procesador para ser analizadas posteriormente por las instrucciones de brinco condicional. Estas instrucciones soportan direccionamiento inmediato y por registro.

Instrucciones de brincos condicionales.

Las instrucciones de brinco condicional son:

BEQ (Branch EQual). Realiza el brinco si los operandos colocados en la instrucción de comparación son iguales.
BNEQ (Branch Not Equal). Realiza el brinco si los operandos colocados en la instrucción de comparación son diferentes.
BLT (Branch Less Than). Realiza el brinco si el primer operando colocado en la instrucción de comparación es menor que el segundo operando.
BLET (Branch Less Equal Than). Realiza el brinco si el primer operando colocado en la instrucción de comparación es menor o igual que el segundo operando.
BGT (Branch Greater Than). Realiza el brinco si el primer operando colocado en la instrucción de comparación es mayor que el segundo operando.
BGET (Branch Greater Equal Than). Realiza el brinco si el primer operando colocado en la instrucción de comparación es mayor o igual que el segundo operando.

AUTOR: VICTOR HUGO GARCÍA ORTEGA

Las instrucciones de brinco condicionales revisan el valor de las banderas almacenadas en el registro de estado del procesador y ejecutan el salto en caso de que la condición que verifican sea verdadera.

Otras instrucciones.

En esta categoría tenemos a la instrucción NOP (No OPeration). Esta instrucción no tiene ninguna función en el procesador, no hace nada, lo único que hace es consumir un ciclo de reloj. Algunas veces el “No hacer nada” es muy útil. En el caso de los procesadores esa instrucción es usada para generar retardos en los programas, eso ayuda para comunicarnos con periféricos que son lentos.